SRC TR 87-180

NASA/USRA Design Project: Computer Graphics Group, Spring, 1987

by

R. Byrne

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
JUN 1987		**00-00-1987 to 00-00-1987**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
NASA/USRA Design Project: Computer Graphics Group, Spring, 1987	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Fairchild Space Company,Germantown,MD,20874**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Much progress has been made by the Computer Graphics Group of the NASA/USRA Design Team at the University of Maryland during the 1986-87 school year. Under the direction of Dr. P. S. Krishnaprasad, the following milestones have been reached: 1) The graphical simulation for the space station now includes a simple solar system (sun, earth, moon) that contains the space station and its associated free-flying robot, the Orbital Maneuvering Vehicle (OMV). 2) The Space Station is now displayed directly from its IGES description, so that future changes in configuration may be easily handled; 3) Co-orbiting with the station is the OMV, which employs fourteen degrees of freedom (six for its body, and four for each of two arms). 4) The earth is now displayed with an outlined drawing of its continents, so that orbital dynamics problems may be simulated such as the rendezvous problem. 5) A Dynamical Simulation of the Two Body Problem has been implemented. Using a VAX 11/785 for the numerical routines and the IRIS 2400 for the display device, the simulation makes use of the 'rsh' mechanism of UNIX 1 for distributing the processing tasks. 6) Many new menu options are available which give users added flexibility. New code under development will create a versatile simulation environment complete with smart buffers, nested menus, and a windowing environment. This will enable users to change simulation parameters for simulation modules running on a different host in the distributed processing network, and allow users to see the new displayed results, without terminating the display program.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**17**	

NASA/USRA Design Project: Computer Graphics Group, Spring, 1987*

Russell Byrne
Fairchild Space Company
Germantown, MD

June, 1987

Abstract

Much progress has been made by the Computer Graphics Group of the NASA/USRA Design Team at the University of Maryland during the 1986-87 school year. Under the direction of Dr. P. S. Krishnaprasad, the following milestones have been reached: 1) The graphical simulation for the space station now includes a simple solar system (sun, earth, moon) that contains the space station and its associated free-flying robot, the Orbital Maneuvering Vehicle (OMV). 2) The Space Station is now displayed directly from its IGES description, so that future changes in configuration may be easily handled; 3) Co-orbiting with the station is the OMV, which employs fourteen degrees of freedom (six for its body, and four for each of two arms). 4) The earth is now displayed with an outlined drawing of its continents, so that orbital dynamics problems may be simulated, such as the rendezvous problem. 5) A Dynamical Simulation of the Two Body Problem has been implemented. Using a VAX 11/785 for the numerical routines and the IRIS 2400 for the display device, the simulation makes use of the 'rsh' mechanism of UNIX [1] for distributing the processing tasks. 6) Many new menu options are available which give users added flexibility. New code under development will create a versatile simulation environment complete with smart buffers, nested menus, and a windowing environment. This will enable users to change simulation parameters for simulation modules running on a different host in the distributed processing network, and allow users to see the new displayed results, without terminating the display program.

*This work was supported in part by NSF Grant #85-00108, AFOSR-URI Grant #AFOSR-87-0073, by a Design Project Grant from NASA through the Universities Space Research Association and by Fairchild Space Company.

[1] UNIX is a trademark of AT&T Bell Laboratories

1 Introduction

As described in [5], during the 1985-86 school year, under the guidance of Dr. P. S. Krishnaprasad, students at the University of Maryland at College Park designed a Mobile Remote Manipulator System (MRMS) for the Space Station. For testing purposes, a MRMS simulator was created. The simulator is graphic, running on an IRIS 2400 Graphics Workstation.

By the end of the school year, the graphic simulation depicted the MRMS moving up and down a keel of the Space Station. The motion was controlled either by users' commands input via a mouse and popup menus, or by an obstacle avoidance algorithm running on a larger computer (e.g. a VAX 11/785 or a Pyramid 90x). Users could view an obstacle avoidance simulation from any vantage point, and change the vantage point during the simulation.

This year Dr. Krishnaprasad has assembled a new group of students who have added many new features to the simulation, and are busy at work finishing others. These features are summarized in the following sections.

2 An Overview of the Display Module

Currently, the display module consists of 38 functions, written in C language. The module may be fed input in one of two formats, or may execute without data input. One of the formats is used for obstacle avoidance/path planning data, described in an earlier paper, and the other is used for the OMV simulation, described below.

The user commands the module by pressing a mouse button, which causes a menu to 'pop up' around the current cursor location. Keeping the button depressed, the user then moves the mouse vertically to select one of the menu entries. The selection is completed when the mouse button is released.

The module consists of roughly 2000 lines of C code. The design is summarized in the following two sections.

2.1 Function Hierarchy

After initialization, the main function enters an endless loop. The mouse is checked for activity, and if a button has been depressed, popup() is called. Popup() returns the menu entry that the user has selected, and this integer is used for a switch statment that deals with each possibility.

If no button has been depressed, the cursor location controlled by the mouse is used to update the variable being changed. For instance, if the user selects 'Move Base', the *velocity* of the base of the MRMS is changed linearly with respect to the distance between the mouse location and the location of the mouse immediately after the previous menu selection.

Another switch statement is used to update the variables corresponding to the various menu entries. Each case of the switch statement updates a variable, and calls a function to 'make' the object with the new parameters.

The end of the loop contains a call to drawall() which simply calls all of the objects.

The objects are drawn on the screen using the 'doublebuffer' mode of the IRIS. Here, two buffers are used. One is used for the current display, and one for the next display. First the next display buffer is cleared, then it is filled with the objects to display in the next instant, and finally the current and next display buffers are switched. This way, the user doesn't see the screen blinking, as would be the case if only one buffer were used.

A basic idea of the complexity of the module may be gained by considering Figure 1. This figure indicates which functions call which others. Since some functions are called by more than one other function, certain functions appear more than once in the figure.

2.2 Object Hierarchy

The objects were created from graphics primitives provided by the IRIS workstation, such as move(x,y,z), and draw(x,y,z) to form vectors. These objects are then called by higher level objects, so that an object hierarchy evolved.

For example, the SOLARSYSTEM is made by calling the SUN and EARTH-MOON objects, with the SUN object calling the GLOBE object, and the GLOBE object calling the AXISOBJ object.

The full hierarchy is detailed in Figures 2 and 3.

3 Menu Options

Until the windowing environment is integrated with the rest of the package, one large menu is used to control everything. It has four sections, one for viewing commands, one for MRMS commands, one for OMV commands, and one for setting or reseting flags. The menu is shown in Figure 4.

The viewing commands may be given at any time, allowing the user to change the viewpoint to verify correct operation of the manipulators being displayed. The choices are *Zoom*, *Pan x*, *Pan y*, and *Stop moving*.

The MRMS commands are used to control the five degrees of freedom of the MRMS. These are base translation, arm yaw, arm pitch, elbow pitch and wrist pitch.

The OMV commands are used to control the 14 degrees of freedom of the OMV. These are x, y and z location; x, y and z rotation, and the same yaw and pitch freedoms mentioned above for the MRMS for each arm of the OMV.

In the last section of the menu, the user may change the value of several flags used in the program. One toggles between displaying everything and displaying

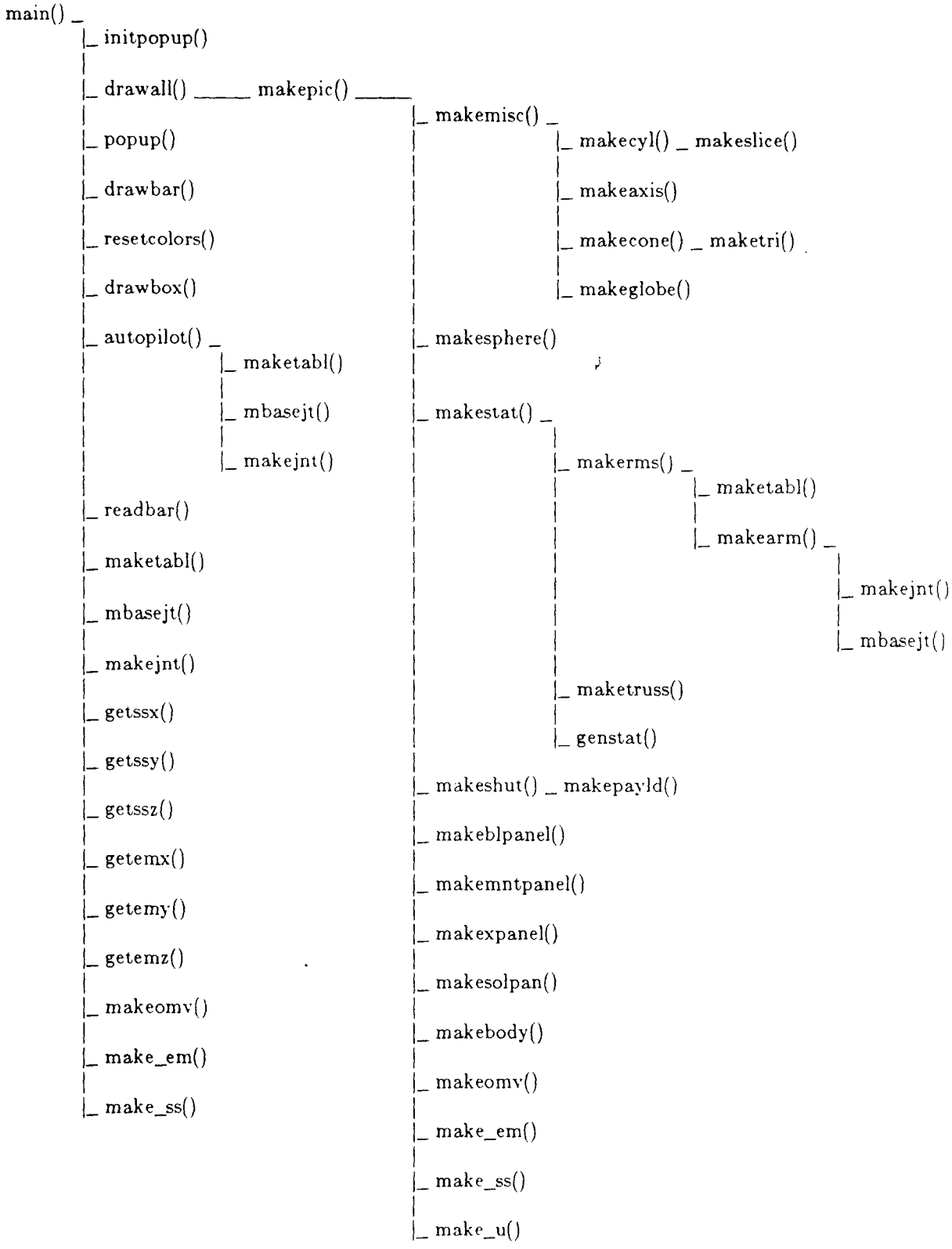

Figure 1: Function Hierarchy

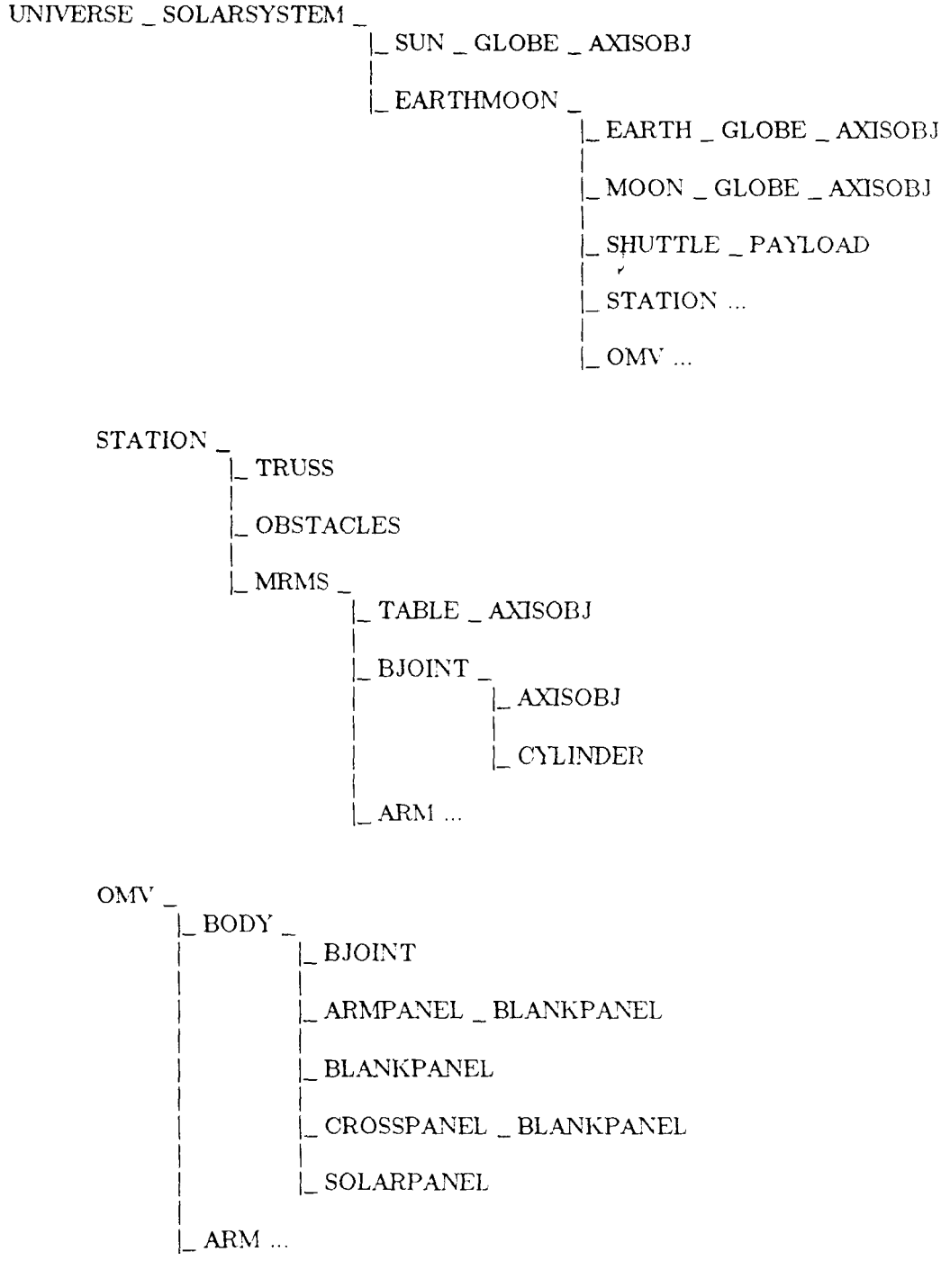

Figure 2: Object Hierarchy

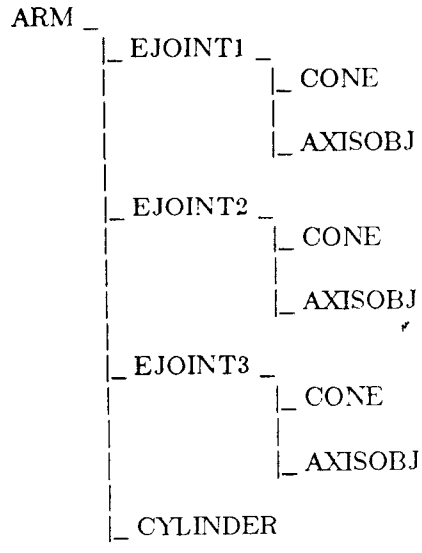

Figure 3: ARM Hierarchy

only the OMV. Another toggles between displaying the solar system from a reference frame attached to the space station and one that is inertial. Another toggles between carrying out the orbital calculations and omitting the same, and the last two are used to start accepting data from one of two possible formats from external files rather than from the mouse commands.

4 The Solar System

One of the first tasks accomplished this school year was the creation of a simple solar system consisting of the sun, earth, moon and space station.

As mentioned above, two reference frames are available: one that is inertial, and one that is fixed to the space station. The orbit of the space station around the earth and the rotation of the earth are presently being simulated, and it is a simple addition to add the orbit of the moon about the earth, and the earth about the sun. The designers of the solar system felt that these non-essential orbits would impose an unacceptable burden on the IRIS 2400. The axis of rotation of the Earth is inclined 23 degrees from its orbit plane, closely modeling the physical motions.

Figure 5 shows a view from the the inertial coordinate frame, with the arrow pointing to the space station. Figure 6 shows the view from the space station reference frame. In this figure, one may see the OMV, MRMS, Space Station, Earth and Sun.

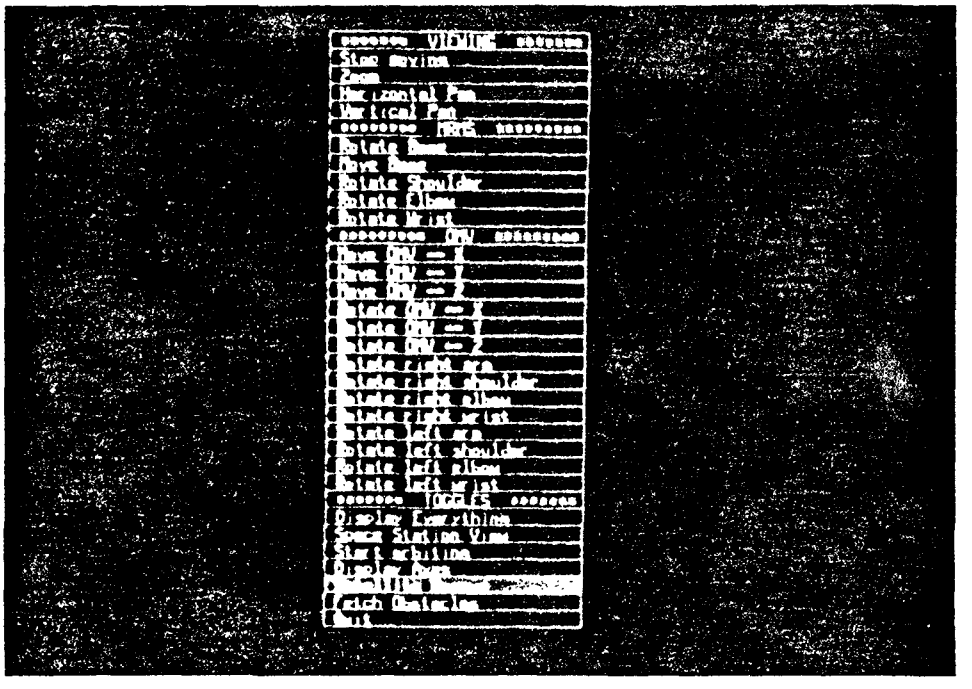

Figure 4: The Menu

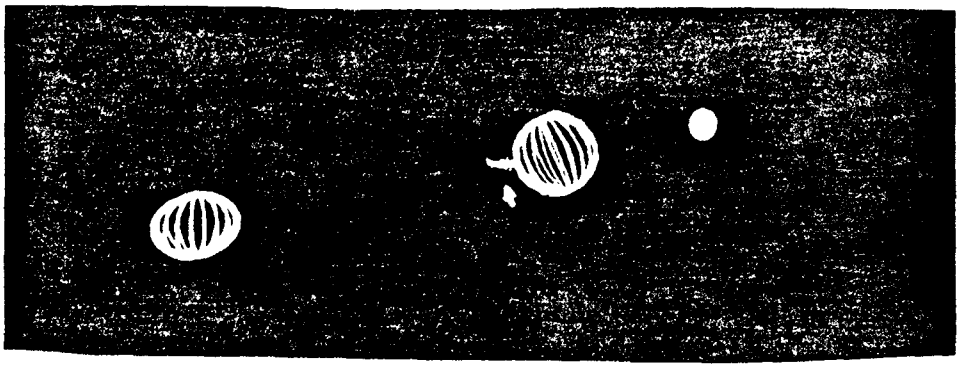

Figure 5: The Inertial Reference Frame

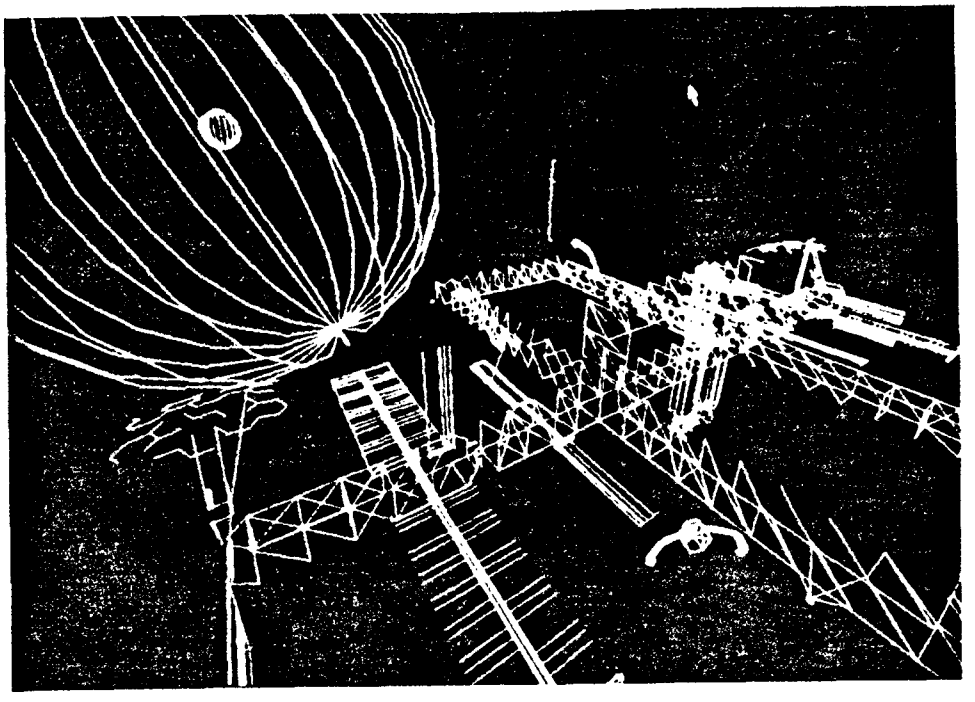

Figure 6: The Space Station Reference Frame

5 The Worldmap

The Worldmap object is a point-by-point, tenth-degree plot of the outlines of the continents. The data was obtained from the Goddard Space Flight Center's computers and was converted and packed to provide a reasonable display without creating a great deal of overhead for the IRIS. Various data sets and plotting techniques were tested to obtain this version.

The data was originally stored as latitude and longitude points in a series of blocks, where one block consisted of 2 to 48 points that could be connected to form a section of an outline. It was decided that an object created by connecting the points would put too big of a demand on the simulation, and a point-model seemed adequate from the Space Station's orbit. Two data sets, one high resolution and one low resolution, were down-loaded to a magnetic tape. The data was then up-loaded onto the VAX and transferred to the IRIS where it was processed.

The IRIS does not have a spherical graphing routine. Thus, the latitude and longitude points had to be mapped into a rectangular coordinate system.

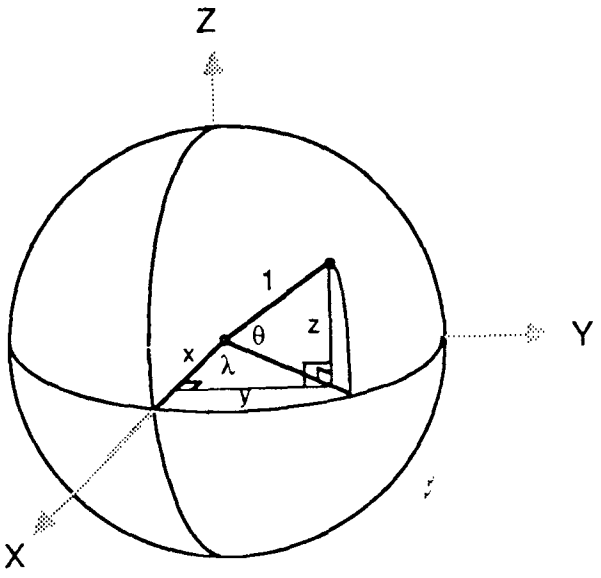

Figure 7: Longitude, Latitude to Rectangular Conversion

The conversion is illustrated in Figure 7, and the following equations:

$$\begin{aligned} x &= cos(\lambda)cos(\theta) \\ y &= cos(\lambda)sin(\theta) \\ z &= sin(\lambda) \end{aligned} \tag{1}$$

where the Earth's radius is normalized to 1.0 for convenience, λ is the latitude, and θ is the longitude.

The converted, low-resolution file contained over 15,000 points spanning 355K of disk space. To reduce the amount of data to be read, the data was packed in binary format, using the write statement in C language. The packing had to be done on the IRIS, because the low-level read and write statements access individual bytes in different orders on the VAX and IRIS (one machine is byte-backwards, the other is byte-forwards). The new, streamlined version takes only half as much space at 180K. The high-resolution model was discarded as being too large for our purposes.

The low-resolution packed file not only conserved disk space, but more importantly, it reduced the time necessary to create the Worldmap object (basically a 'load'), and it kept the object size to a minimum. The resulting object doesn't appreciably slow down the rest of the graphics.

At present, the Worldmap representation is adequate but needs slight improvements. The IRIS 2400 has only 16 bitplanes. As such, the simulation is displayed in 'wire-frame' format. Land masses are outlined, so the opposite side of the world is superimposed on the viewed side. This is especially confusing in this case. A possible solution to this problem is to blank out the opposite side of the world by placing a black disk normal to the line-of-sight in the center of the globe, but since perspective is constantly changing, this is difficult to implement. Another possibility would be to create a solid black sphere located

Figure 8: The Worldmap Object

just inside the Worldmap object by using several black disks (a sphere of revolution). The best solution would be to obtain an IRIS 3130 workstation, which has 32 bitplanes. With this device, all of the objects in the simulation could be displayed in solid form, using depth cuing.

The Worldmap provides a reference system for other objects. Satellites can be seen to pass over certain parts of the world, and tracking stations are much better represented when they are seen with a world map. In addition, it also adds to the realism of the simulation. The current map is shown in Figure 8.

6 IGES Decoding

Since the Space Station is still in its early design stages, the basic configuration (one keel or two, or something else) has still not been firmly decided upon. Thus, each time NASA makes the newest configuration known, the object oriented C code on the IRIS must be converted to reflect the new configuration. This is tedious!

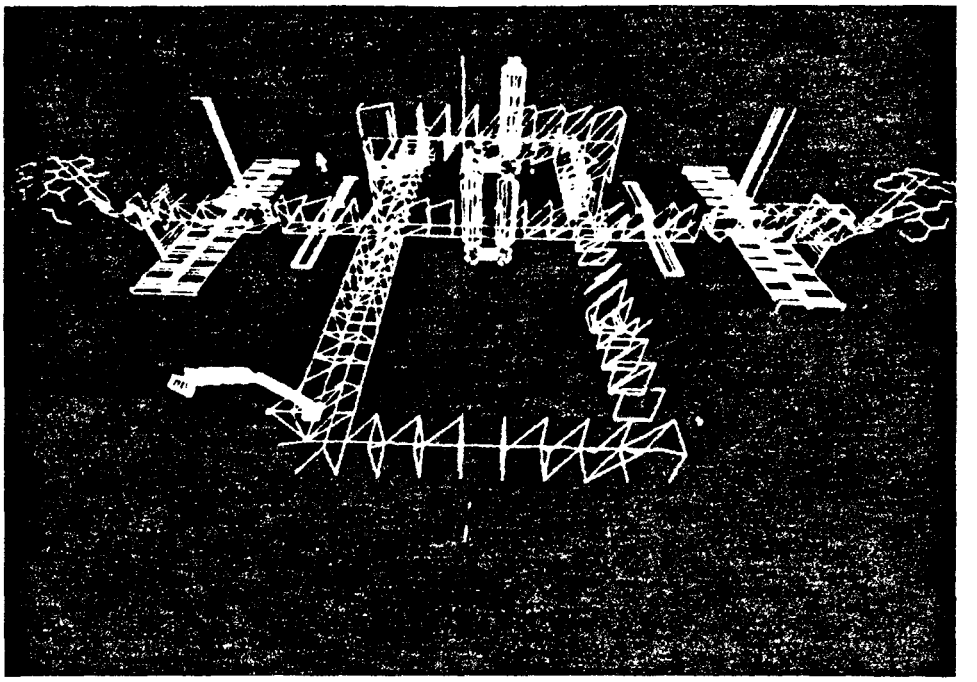

Figure 9: IGES Definition of the Space Station

To avoid the necessity of changing the C code each time the Space Station configuration changes, the Space Station object is now created from its IGES[2] description directly. When the Space Station changes shape on NASA's drawing board, all that is necessary is to obtain the IGES description on a magnetic tape, upload it onto the IRIS, and run the 'IGES Compiler'.

The 'IGES Compiler' reads the IGES description, extracts the pertinent information therein, and then converts vector and curve descriptions into the syntax that the IRIS understands. As an example of the complexity of the Space Station object, the following are clearly visible in Figure 9: the Solar Dynamic Collector/Receivers and Thermal Control Radiators; the Photovoltaic Electrical Power Arrays and Thermal Control Radiators; the Pressurized Habitation, Laboratory and Logistics Modules; and of course, the Mobile Remote Manipulator System.

7 The OMV

As described in [1], the graphical object representing the Orbital Maneuvering Vehicle has been created. It is shown in Figure 10. All of the dimensions of the object are defined by three quantities: the length and width of the side panels

[2]Initial Graphics Exchange Specification (IGES) is a specification of a set of information structures, used for compatible exchange of product definition data used by various CAD/CAM systems [6].

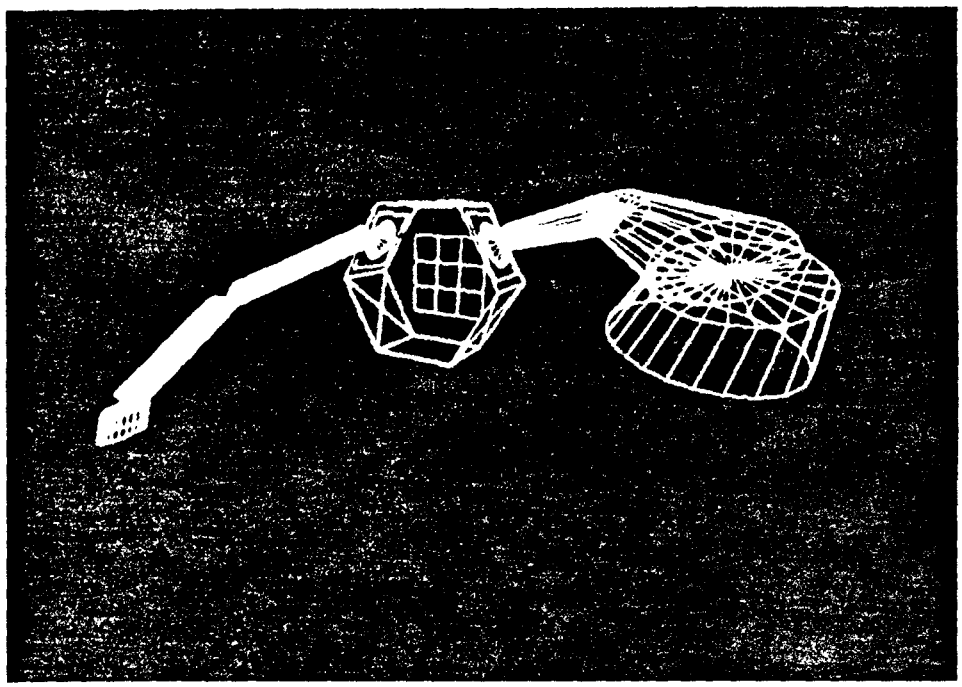

Figure 10: The OMV

that make up the OMV's body, and the ratio of the size of the OMV's arms to the size of the MRMS. Note that the arms of the OMV are simply scaled down versions of the MRMS without the movable base. This demonstrates the beauty of object oriented graphics.

The object on the back of the OMV represents a solar array. Later, a simplified arm will be added to the bottom of the OMV. This will be used as a grappling tool, to hold on to the work-site. In addition, future plans call for adding CCTV cameras and various end-effectors.

As described above, the OMV has 14 menu-selectable degrees of freedom. Three control the position of the body, three control the attitude, and there are four degrees of freedom in each arm. With the current menu as a user interface, only one of these freedoms may be exercised at a time. Since most tasks require more than one degree of freedom to be active concurrently, another input method was required.

The Display Module may be fed data into its standard input stream, and this can be used to control the motion of the OMV. The data format consists of records of 14 values split into two lines. The first line of an input file contains the six values for the x, y and z location of the body, and the roll, pitch and yaw body coordinate frame rotations. The next line contains the four values defining the position of the left arm and four values for the right arm.

A simple 'Satellite Rescue' has been written, employing this data input method. The OMV leaves its home position near the Space Station, rendezvous

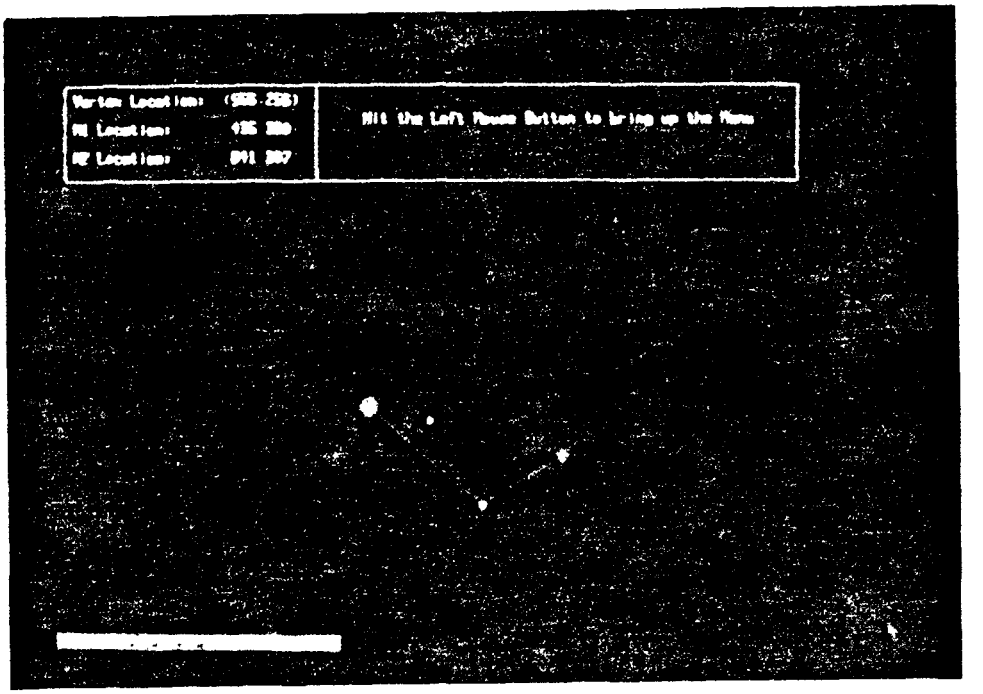

Figure 11: The Two Body Simulation

with the target satellite, captures it, and returns it to the Space Station. The rendezvous task and the capture task will be focused upon in future research, leading to a dynamical simulation of both.

8 Dynamical Simulation of the Two Body Problem

As described in [7], a dynamical simulation of the two rigid body problem has been designed. The equations of motion for the system shown in Figure 11 were derived using the Hamiltonian (rather than the classical Euler) approach. These equations are solved using the Runge Kutta method.

The simulation assumes that no external forces act on the two body system. Also, the joint connecting the two bodies is assumed massless. In the absence of external forces, a torque applied to the joint will result not only in relative angular displacement between the two bodies, but also in rectilinear displacement of the joint itself. This is due to conservation of the system center of mass.

The input for this simulation includes the energy in the system, the simulation update time step, total simulation time, print frequency, body masses, inertias, and semimajor axes. The bodies are thus modeled as generalized two-dimensional shapes, but displayed as point-masses.

The simulation is started by defining the inputs in a file resident on the IRIS

workstation. Then the dynamical simulation module is started on the VAX, using the 'rsh' mechanism of UNIX [3]. The input for the module running on the VAX is taken from the IRIS, and the output is then sent to the display module on the IRIS.

The display consists of the depiction of the two body system, the center of mass of the system (which remains constant), and a trail of dots traced by the joint of the system. At the top of the display, updating at a ridiculously frequent rate, are the coordinates of the two bodies and the joint.

Future work here is the most important of all the work described thus far. Currently, this simulation may be used in the OMV simulation, if all but one of the OMV's arm joints are locked in place. The next few simulations to be performed are the three and five body cases. With a five body dynamical simulation, much will be learned about the OMV's capture task.

Finally, to realistically simulate the OMV's tasks, these simulations must make the leap from two to three dimensions. When this is done, Robotics and Attitude Control engineering students will have a very useful test bed for their designs.

9 Interprocess Communication

As described in [2], [3] and [5], the MRMS simulation was split up into three tasks: The Command System, The Control System and the Display System. These tasks were in turn divided up into functional modules. The modules communicate with each other through standard input and output streams.

Essentially, two pipelines of modules are set up; one for communicating from the Command System to the Control System; and one for communicating from the Control System to the Display System. Within a given pipeline, a given module is in an endless loop checking for data on the input stream and executing its task. If data is encountered, the module checks to see if the command and/or data is intended for the given module or some other module. In the former case, the given module processes the command, perhaps changing some parameters, and resumes its task. In the later case, the given module copies the command and/or data to its standard output so that the modules downstream may do the same check.

Straddling these two pipelines is a 'smart buffer' or 'tape recorder' module. This may be used for reviewing certain portions of a simulation, or pausing a simulation so that a better viewpoint may be obtained.

With these communications tools, it is unlikely that the processes comprising the simulation will need to be terminated. Thus, users unfamiliar with the operating systems involved may still be able to use the simulation (i.e., typical users will not need to know how to start the simulation).

[3] UNIX is a trademark of AT&T Bell Laboratories.

10 The MEX Windowing Environment

In [4] Sinha explains the implementation of the Multiple EXposure (MEX) windowing environment. Here the features are discussed from the users' point of view.

Since the IRIS was intended to be used as both the command input device and the display device, some sort of windowing or split-screen tool had to be developed. The IRIS provides support for a very general windowing tool.

In the environment Sinha has created, users not only have different windows for such things as the display of the simulation, parameter input, simulation control, etc., but they may arrange the windows (shape and location) as they wish. Thus, users may use the entire screen for the display, and then shrink the display, enlarge the parameter input window, and change simulation parameters.

Inside the windows 'pull-right' menus pop up when mouse buttons are pressed. This makes it easy for users to activate the more common menu options (i.e., they reside on the first level menu), and yet the menus are kept to a manageable size. In addition, the choice of menu options changes to reflect the valid choices for any given simulation configuration.

In the future, this environment may be used to create areas for parameter display in numeric and/or graphic form, three-view images of given objects, and whatever else is of interest. Since the size of each window can be changed dynamically, the available space on the screen in not of concern.

11 Conclusion

Now that the interprocess communications and windowing environment work are done, the next task is to increase the complexity of the dynamic modeling. The two body results may be used for the three-dimensional OMV simulation if all but one of the OMV's joints are locked. One planned simulation will let the users specify a joint to be moved, and the others will be locked. Then the OMV's initial center of mass will be calculated, and displayed. Next, following the users' input, motor commands will be sent to the joint, and the resulting motion of the OMV displayed will conserve this center of mass.

The level of complexity will rise from the two body in the plane to the eight body (OMV body, three links for each of two arms, and a load) problem in space. Eventually, a very useful tool for testing realistic space-based robotic and attitude control schemes for the OMV and the Space Station will have evolved. In addition, future work will include orbital rendezvous simulations, and a space station construction simulation.

Acknowledgments

The author is grateful to Dr. P. S. Krishnaprasad for his many useful suggestions; Kenneth Rachlin for his assistance with the Worldmap documentation, and to the reviewer for his comments.

References

[1] Russell Byrne. *Architecture of MRMS Simulation: The Orbital Maneuvering Vehicle.* Intelligent Servosystem Laboratory, Systems Research Center, The University of Maryland, College Park, MD, 1987. In Preperation.

[2] Velu Sinha. *Architecture of MRMS Simulation: Distributing Processes.* Intelligent Servosystem Laboratory TR8725, Systems Research Center, The University of Maryland, College Park, MD, December 1986.

[3] Velu Sinha. *Architecture of MRMS Simulation: Interprocess Communciations.* Intelligent Servosystem Laboratory, Systems Research Center, The University of Maryland, College Park, MD, 1987. In Preperation.

[4] Velu Sinha. *Architecture of MRMS Simulation: The MEX Windowing Environment.* Intelligent Servosystem Laboratory, Systems Research Center, The University of Maryland, College Park, MD, 1987. In Preperation.

[5] Velu Sinha. *The Mobile Remote Manipulator System Simulator.* Intelligent Servosystem Laboratory TR8724, Systems Research Center, The University of Maryland, College Park, MD, December 1986.

[6] Bradford M. Smith, et al. *Initial Graphics Exchange Specification (IGES), Version 2.0.* National Bureau of Standards, Center for Manufacturing Engineering, Washington, DC 20234, February 1983.

[7] N. Sreenath and Russell Byrne. *Architecture of MRMS Simulation: Dynamic Modeling of the Two Body Problem.* Intelligent Servosystem Laboratory, Systems Research Center, The University of Maryland, College Park, MD, 1987. In Preperation.